Cartographic documents for web modeling and representation of indoor mapping with interactive environments

[Extended Abstract] *

Ben Trovato†
Institute for Clarity in Documentation
1932 Wallamaloo Lane
Wallamaloo, New Zealand
trovato@corporation.com

G.K.M. Tobin‡
Institute for Clarity in Documentation
P.O. Box 1212
Dublin, Ohio 43017-6221
webmaster@marysville-ohio.com

Lars Thørväld§
The Thørväld Group
1 Thørväld Circle
Hekla, Iceland
larst@affiliation.org

Lawrence P. Leipuner
Brookhaven Laboratories
Brookhaven National Lab
P.O. Box 5000
lleipuner@researchlabs.org

Sean Fogarty
NASA Ames Research Center
Moffett Field
California 94035
fogartys@amesres.org

Charles Palmer
Palmer Research Laboratories
8600 Datapoint Drive
San Antonio, Texas 78229
cpalmer@prl.com

ABSTRACT

PUT THE ABSTRACT HERE

Categories and Subject Descriptors

H.4 [**Information Systems Applications**]: Miscellaneous; D.2.8 [**Software Engineering**]: Metrics—*complexity measures, performance measures*

General Terms

Theory

Keywords

ACM proceedings, LaTeX, text tagging

1. INTRODUCITON

...

*A full version of this paper is available as *Author's Guide to Preparing ACM SIG Proceedings Using LaTeX2$_\epsilon$ and BibTeX* at `www.acm.org/eaddress.htm`

†Dr. Trovato insisted his name be first.

‡The secretary disavows any knowledge of this author's actions.

§This author is the one who did all the really hard work.

The remainder of the paper is organized as follows. In Section II is provided an overview of the state of the art. Section III is devoted to describe the novel cartographic document proposed, while section IV introduces the underlying mathematical structure. Section V reports about the tools and instruments developed specifically for the web, focusing on software architecture, implemented algorithms and real applications. Section VI describes one use case of both document format and software tools. Finally Section VII proposes some conclusive remarks and future developments.

2. STATE OF THE ART

PUT THE STATE OF THE ART HERE

2.1 GeoJSON

GeoJSON is a format for encoding a variety of geographic data structures. GeoJSON supports the following geometry types: `Point`, `LineString`, `Polygon`, `MultiPoint`, `MultiLineString`, and `MultiPolygon`. Lists of geometries are represented by a `GeometryCollection`. Geometries with additional properties are `Feature` objects. And lists of features are represented by a `FeatureCollection`. GeoJSON%20spechttp://gespec.html

GeoJSON is good for geographic mapping application, but not for indoor application. There is a GeoJSON variant suitable for indoor app.

2.2 Experiences on Indoor JSON

`IndoorJSON` is a GeoJSON variant used by indoor.io toolset to define indoor maps. IndoorJSON may consist of any number of Features and/or FeatureCollections. All Features are interpreted similarly regardless of their grouping into nested FeatureCollections. IndoorJSON supports all GeoJSON geometry types.

3. ADVANCES ON CARTOGRAPHICS DOCUMENT STANDARDS: HIJSON

HIJSON (**H**ierarchical **I**ndoor **JSON**) is a GeoJSON variant. A HIJSON document reveals at least three major enhancements above the actual state of the art in indoor cartographic documents:

1. Hierarchical structure
2. Metric local coordinate System
3. Semantic extensions

3.0.1 Hierarchical structure

Unlike other formats like GeoJSON or IndoorJSON, HIJSON organizes its elements in a hierarchical structure, where every element represent a potential container for other elements. This structure allows a clear and logical organization of the elements inside the structure, and at the same time make it possible to use a relative, local, metric coordinates system.

3.0.2 Metric local coordinate System

In GeoJSON all the positions are expressed in geographical coordinates (usually WGS84). Although this can be useful for outdoor geographical representations, it is not the best solution for indoor descriptions. In HIJSON all the coordinates are expressed in a relative system based on the hierarchical structure. The shape of all elements is described starting from origin, and then two vectors (translation and rotation) describe the position relative to the origin of the parent element. By this way it is possible to describe the position of a piece of furniture by specifying its distance from the origin of the room, that is obviously more convenient than describing its geographical coordinates. Another advantage is represented by the adoption of a metric reference. A recursive process that computes intermediate transformation matrixes can then produce a standard GeoJSON representation, that can be visualized on any standard viewer.

3.0.3 Semantic extensions

Every HIJSON Element has a property that describes its class. This information allows the adoption of semantic extensions by the software that manipulates the HIJSON data. In the Javascript library developed to manage HIJSON documents, different classes are instantiated to represent HIJSON Nodes, which acts differently by the adoption of polymorphic methods. Ino rder to extend the possibilities in representation and interaction, it is sufficient to define new classes that reflects new categories of HIJSON Elements.

4. LAR: THE UNDERLYING MATHEMATICAL STRUCTURE

5. IMPLEMENTED WEB TOOLS

A set of web based instruments has been developed allowing to deal with the HIJSON document previously described. Tools are written in *JavaScript* language, using *Node.js* and in particular *Express.js* as backend framework, and exploiting the power of WebSocket protocol through the *Socket.io* library.

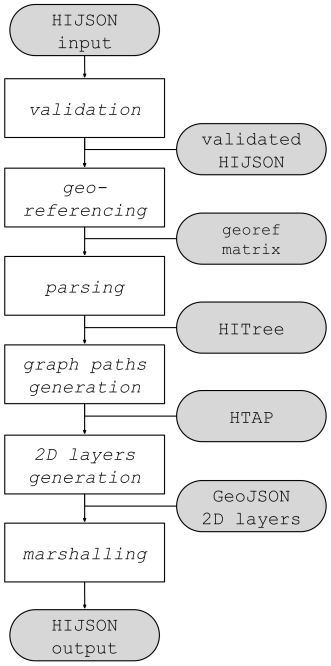

Figure 1: HIJSON processing pipeline

5.1 Architecture

The overall architecture of the tools are, as for the vast majority of the web based application, inherently *client/server*.

5.1.1 HIJSON processing pipeline

Each time a new HIJSON document is submitted to the server, it is passed through the **HIJSON processing pipeline**, where it is subjected to a sequence of preliminary transformations.

The application of the transformation pipeline has a double aim. The first one consists in generating the graph of valid paths between all the interesting HIJSON elements. The second aim is the generation of one *GeoJSON* document for each story of the building described by the HIJSON document. In this way any connected client can be provided with a bidimensional plant for each level of the building that it can visualize through any compliant GeoJSON viewer.

HIJSON processing pipeline (as pictured in figure ...) is composed by 6 elaboration stages. In the following are detailed operations excecuted by each stage, which are, in the order: *validation*, *georeferencing*, *parsing*, *graph paths generation*, *2D layers generation*, *marshalling*.

1. [**validation**] - The first one is a validation stage. In order to begin with the effective transformations the input HIJSON document must be compliant with the rules defined in (AGGIUNGERE REF TO PARAGRAFO REGOLE DI VALIDITA'). In the case the validation stage fails, processing aborts and do not continue to following stages. If the stage success, the output for the next stage is a validated HIJSON. 2. [**georeferencing**] - 3. [**parsing**] - The parsing stage, takes the validated HIJSON as its input, that as illustrated before can be thought of as a list of HIJSON Elments, parses them and produce an istance of HITREE, which is an object in memory representing the tree hierarchical structure of the building described into the HIJSON. 4.

[**graph paths generation**] - The fourth stage is in charge of THE generatio of the graph paths. This aim is accomplished according to the algorithm described in (AGGIUNGERE RIFERIMENTO A #### Automatic generation of valid paths). The graph paths will be useful afterwards to coumpute valid paths from couple of point of interest on the graph. Once the graph paths has been computed, the input HITREE is augmented with paths information, becoming what has been called an HTAP (HITREE Augmented with Paths). Augmentation always takes place as leaf nodes added as children of a specific (e.g. "room") level. 5. [**2D layers generation**] - The fifth state is the generation of geoJSON layer. For each level, the system generates one geoJSON layer that will be use for the creation of 2D map. Each layer contains the children of 'level' node in the HITREE. Every class contains a boolean value that is use to choose which class will be a part of geoJSON layer. Every element has a geographical coordinates calculated by the transformation matrix with regard to the local coordinates of the HIJSON element. 6. [**marshalling**] -

5.1.2 Client

When a client connects to the server, it receives the HIJSON input files, the ready-to-use GeoJSON layers and the weighted adjacency matrix of the grapth paths. A very short pipeline of processing is performed by the client, composed by: 1. [**Parsing**] - Like for the Server-side processing, the HIJSON Elements in the input files are processed and tranformed in HIJSON Nodes, linked together in a hierarchical tree structure. 2. [**3D Model generation**] - Unlike HIJSON Elements (that are simple Javascript objects), HIJSON Nodes are instances of specific classes, representing a particular category of element in the building. Through a polymorphic method, each node generates a Three.js 3D model of its entity, that is used to assemble a complete 3D Model of the building. The similarities in HIJSON hierarchical structure and Three.js scene graph allow this process to be performed with little effort.

5.2 Algorithmics

5.2.1 Automatic generation of valid paths

One of the core functionalities of the pre-processing server stage is the generation of a graph of valid paths trough the entire building, that can be used to calculate directions between two given nodes (see Crossfloor user navigation paragraph). Taking advantage of the hierarchical structure of the representation, the problem is splitted in many sub-problems represented by the computation of the sub-graphs relative to each room. The sub-graphs are then linked together trough doors. The entire process (as shown in figure), is composed by 4 phases: 1. Computation of the walkable area of the room; 2. Triangulation of the walkable area; 3. For each triangle side completely internal to the area, its midpoint is used to represent a graph node; 4. Nodes relative to the same triangle are then linked together, and the doors are linked to the nearest node in the room.

After that, the objects in the room are linked to the graph nodes as well, giving the possibility to calculate paths to specific elements of interest. Given the nature of this process, the paths calculated may not be absolutely optimal, but this strategy ensures a good ratio between extension of exploration and number of graph nodes.

5.3 Applications

5.3.1 IoT monitoring

Every element in 2D map or in 3D model is interactive and the user can ask for information. In every class there is a method that gets information and send that to the client. The modularity of *nome del software* permits to show particular information with regard to the object. There are two groups of objects: simple and smart. If the object is smart, it can send data in real time through its sensor (e.g. if the object is a thermostat, the user can see the temperature in the room and can turn on, or off, the heating). If the object isn't smart, the system can show static information (e.g. for fire Extinguisher, the system shows the last date of checking).

5.3.2 Realtime access monitoring

The system can be used for access monitoring. On 2D map will be a marker for each person in the building, whereas on the 3D model will be a 3D model of user. With an appropriate system of indoor localization, every person sends its position in real time. The 2D map and the 3D model is automatically refreshed. The user can ask for information about person, in function of the use of the system.

5.3.3 Crossfloor user navigation

With the weighted adjacency matrix, the user can choose two nodes that represent respectively the start and the end point. The system calculates the optimal path by Dijkstra's algorithm and shows this with a polyline in 2D map. To pass through different floors, the path leads to stairs or elevators. All the nodes of the same elevator are connected among them and their distances is set to 0. Otherwise the connections through the stairs are characterized by the nodes of two different levels; their weight of this portion of path is set to the distance between these nodes. Therefore the subgraph that characterized stairs or elevator is complete.

6. USE CASE

`C3D.js`

7. CONCLUSIONS

We presented HIJSON a GeoJSON extension for indoor mapping

8. ACKNOWLEDGMENTS

This section is optional; it is a location for you to acknowledge grants, funding, editing assistance and what have you. In the present case, for example, the authors would like to thank Gerald Murray of ACM for his help in codifying this *Author's Guide* and the **.cls** and **.tex** files that it describes.

APPENDIX

A. HEADINGS IN APPENDICES

The rules about hierarchical headings discussed above for the body of the article are different in the appendices. In the **appendix** environment, the command **section** is used to indicate the start of each Appendix, with alphabetic order designation (i.e. the first is A, the second B, etc.) and a title

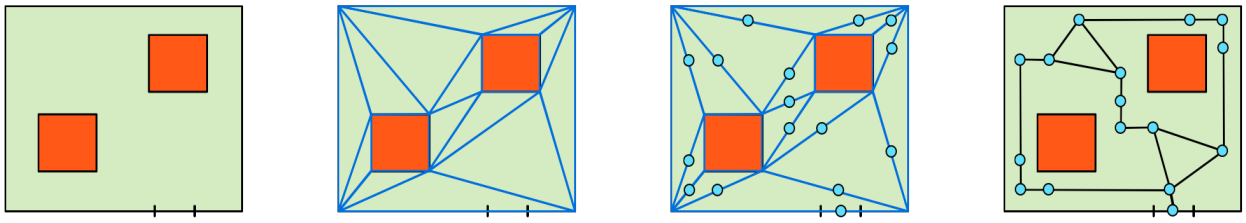

Figure 2: Graph paths generation. (a) detection of obstacles; (b) triangulation of walkable area; (c) identification of graph nodes area; (d) junction of nodes.

(if you include one). So, if you need hierarchical structure *within* an Appendix, start with **subsection** as the highest level. Here is an outline of the body of this document in Appendix-appropriate form:

A.1 Introduction

A.2 The Body of the Paper

A.2.1 Type Changes and Special Characters

A.2.2 Math Equations

Inline (In-text) Equations.

Display Equations.

A.2.3 Citations

A.2.4 Tables

A.2.5 Figures

A.2.6 Theorem-like Constructs

A Caveat for the TEX Expert

A.3 Conclusions

A.4 Acknowledgments

A.5 Additional Authors

This section is inserted by LATEX; you do not insert it. You just add the names and information in the `\additionalauthors` command at the start of the document.

A.6 References

Generated by bibtex from your .bib file. Run latex, then bibtex, then latex twice (to resolve references) to create the .bbl file. Insert that .bbl file into the .tex source file and comment out the command `\thebibliography`.

B. MORE HELP FOR THE HARDY

The sig-alternate.cls file itself is chock-full of succinct and helpful comments. If you consider yourself a moderately experienced to expert user of LATEX, you may find reading it useful but please remember not to change it.